AN ADAPTIVE VARIABLE ORDER QUADRATURE STRATEGY

PAUL HOUSTON AND THOMAS P. WIHLER

Abstract. In this article we propose a new adaptive numerical quadrature procedure which includes both local subdivision of the integration domain, as well as local variation of the number of quadrature points employed on each subinterval. In this way we aim to account for local smoothness properties of the function to be integrated as effectively as possible, and thereby achieve highly accurate results in a very efficient manner. Indeed, this idea originates from so-called *hp*-version finite element methods which are known to deliver high-order convergence rates, even for nonsmooth functions.

1. Introduction

Numerical integration methods have witnessed a tremendous development over the last few decades; see, e.g., [2, 3, 15]. In particular, adaptive quadrature rules have nowadays become an integral part of many scientific computing codes. Here, one of the first yet very successful approaches is the application of adaptive Simpson integration or the more accurate Gauss-Kronrod procedures (see, e.g., [7]). The key points in the design of these methods are, first of all, to keep the number of function evaluations low, and, secondly, to divide the domain of integration in such a way that the features of the integrand function are appropriately and effectively accounted for.

The aim of the current article is to propose a complementary adaptive quadrature approach that is quite different from previous numerical integration schemes. In fact, our work is based on exploiting ideas from *hp*-type adaptive finite element methods (FEM); cf. [4, 6, 12, 13, 20]. These schemes accommodate and combine both traditional low-order adaptive FEM and high-order (so-called spectral) methods within a single unified framework. Specifically, their goal is to generate discrete approximation spaces which allow for both adaptively refined subdomains, as well as locally varying approximation orders. In this way, the *hp*-FEM methodology is able to resolve features of an underlying unknown analytical solution in a highly efficient manner. In fact, this approach has proved to be enormously successful in the context of numerically approximating solutions of differential equations, and has been shown to exhibit high-order algebraic or exponential convergence rates even in the presence of local singularities; cf. [9, 17, 18].

With this in mind, we adopt the *hp*-adaptive finite element strategy for the purpose of introducing a variable order adaptive quadrature framework. More precisely, we propose a procedure whereby the integration domain will be subdivided adaptively in combination with a local tuning of the number of quadrature points employed on each subinterval. To drive this refinement process, we employ a smoothness estimation technique from [6, 22] (see also [12] for a related strategy), which was originally introduced in the context of *hp*-adaptive FEM. Specifically, the smoothness test makes it possible to gain local information concerning the regularity of the integrand function, and thereby, to suitably subdivide the integration domain and select an appropriate number of quadrature points for each subinterval. By means of a series of numerical experiments we demonstrate that the proposed adaptive quadrature strategy is capable of generating highly accurate approximations at a very low computational cost. The main ideas on this new approach together with a view on practical aspects will be discussed in the subsequent section.

2010 *Mathematics Subject Classification.* 65D30,65N30.

Key words and phrases. Adaptive quadrature, *hp*-adaptivity, adaptive Gauss quadrature.

TW acknowledges the financial support by the Swiss National Science Foundation (SNF).

2. An hp-Type Quadrature Approach

2.1. **General Quadrature Rules.** Typical quadrature rules for the approximation of an integral

$$I := \int_{-1}^{1} f(x)\,\mathsf{d}x \tag{2.1}$$

of a continuous function $f : [-1,1] \to \mathbb{R}$, take the form

$$I \approx \widehat{Q}_p(f) := \sum_{k=1}^{p} w_{p,k} f(\widehat{x}_{p,k}), \tag{2.2}$$

where $p \geq 1$ is a (typically prescribed) integer number, and $\{\widehat{x}_{p,k}\}_{k=1}^{p} \subset [-1,1]$ and $\{w_{p,k}\}_{k=1}^{p} \subset (0,2]$ are appropriate quadrature points and weights, respectively. When dealing with a variable number p of quadrature points and weights, we can consider one-parameter families of quadrature rules (such as, for example, Gauss-type quadrature methods); here, for each $p \in \mathbb{N}$, with $p \geq p_{\min}$, where $p_{\min}$ is a minimal number of points, there are (possibly non-hierarchical) families of quadrature points $\widehat{\boldsymbol{x}}_p = \{\widehat{x}_{p,k}\}_{k=1}^{p}$, and weights $\boldsymbol{w}_p = \{w_{p,k}\}_{k=1}^{p}$.

On an arbitrary bounded interval $[a,b]$, $a < b$, a corresponding integration formula can be obtained, for instance, by means of a simple affine scaling

$$\phi_{[a,b]} : [-1,1] \to [a,b], \qquad \widehat{x} \mapsto x = \phi_{[a,b]}(\widehat{x}) = \frac{1}{2}h\widehat{x} + \frac{1}{2}(a+b), \tag{2.3}$$

with $h = b - a > 0$. Indeed, in this case

$$\int_a^b f(x)\,\mathsf{d}x \approx Q_{[a,b],p}(f) := \frac{h}{2}\sum_{k=1}^{p} w_{p,k}(f \circ \phi_{[a,b]})(\widehat{x}_{p,k}),$$

where $f : [a,b] \to \mathbb{R}$ is again continuous. As before, for any specific family of quadrature rules, the corresponding quadrature point families $\boldsymbol{x}_p$ are obtained in a straightforward way by letting $\boldsymbol{x}_p = \phi_{[a,b]}(\widehat{\boldsymbol{x}}_p)$ (with the understanding that $\phi_{[a,b]}$ is extended componentwise to vectors).

Furthermore, the above construction allows us to define composite quadrature rules, whereby the integral of f is approximated on a collection of $n \geq 1$ disjoint (open) subintervals $\{K_i\}_{i=1}^{n}$ of $[a,b]$ with $[a,b] = \bigcup_{i=1}^{n} \overline{K}_i$, i.e.,

$$I \approx \sum_{i=1}^{n} Q_{K_i,p}(f|_{K_i}).$$

In practical applications the subintervals are usually either of uniform size $(b-a)/n$, for sufficiently large n, or alternatively, they are selected adaptively with the aim of resolving the relevant features of the given function f.

2.2. **The Basic Idea: hp-Adaptivity.** Adaptive quadrature rules usually generate a sequence of repeatedly bisected and possibly non-uniform subintervals $\{K_i\}_{i=1}^{n}$, $n \geq 1$, of the integration domain $[a,b]$ (i.e., each subinterval K_i may have a different length h_i), with a prescribed and uniform number p of quadrature points on each subinterval. With the aim of providing highly accurate approximations with as little computational effort as possible, the novelty of the approach presented in this article is to design an adaptive quadrature procedure, which, in addition to subdividing the original interval $[a,b]$ into appropriate subintervals, is able to adjust the number of quadrature points p_i *individually* within each subinterval K_i in an effective way. We note that this idea originates from approximation theory [5,16] (see also [9]), and has been applied with huge success in the context of finite element methods for the numerical approximation of differential equations. Indeed, under certain conditions, the judicious combination of subinterval refinements (h-refinement) and selection of local approximation orders (p-refinement), which results in the class of so-called hp-finite element methods, is able to achieve high-order algebraic or exponential rates of convergence, even for solutions with local singularities; see, e.g. [18]. In an effort to automate the combined h- and p-refinement process, a number of hp-adaptive finite element approaches have been proposed in the literature; see, e.g, the survey article [14] and the references cited therein. In the current article, we pursue the smoothness estimation approach developed in [6,22] (cf. also [12] for a related methodology), and translate the idea into the context of adaptive variable order numerical quadrature.

Starting from a subinterval K_i with p_i quadrature points, we are given a current approximation $Q_{K_i,p_i}(f|_{K_i})$ of the subintegral

$$\int_{K_i} f(x)\,\mathsf{d}x \approx Q_{K_i,p_i}(f|_{K_i}). \tag{2.4}$$

Then, with the aim of improving the approximate value $Q_{K_i,p_i}(f|_{K_i})$, in the sense of an hp-adaptive finite element methodology in one-dimension, we propose two possible refinements of K_i:

(i) h-refinement: The subinterval K_i of length h_i is bisected into two subintervals K_i^1 and K_i^2 of equal size $h_i/2$, and the number p_i of quadrature points is either inherited to both subintervals or, in order to allow for derefinement with respect to the number of local quadrature points, reduced to $p_i - 1$ points. In the latter case, we obtain a potentially improved approximation

$$Q^{\mathrm{h}}_{K_i}(f) = Q_{K_i^1,\max(1,p_i-1)}(f) + Q_{K_i^2,\max(1,p_i-1)}(f) \tag{2.5}$$

of (2.4).

(ii) p-refinement: The subinterval K_i is retained, and the number p_i of quadrature points p_i is increased by 1, i.e., $p_i \leftarrow p_i + 1$. This yields an approximation

$$Q^{\mathrm{p}}_{K_i}(f) = Q_{K_i,p_i+1}(f). \tag{2.6}$$

In case that $p_i = p_{\max}$, where $p_{\max}$ is a prescribed maximal number of quadrature points on each subinterval, we define

$$Q^{\mathrm{p}}_{K_i}(f) = Q_{K_i^1,p_i}(f) + Q_{K_i^2,p_i}(f), \tag{2.7}$$

where K_i^1 and K_i^2 result from subdividing K_i as in (i).

In order to determine which of the above refinements is more appropriate for a given subinterval K_i, we apply a smoothness estimation idea as outlined in the subsequent section. Once a decision between h- and p-refinement for K_i has been made, the procedure is repeated iteratively for any subintervals K_i for which $Q_{K_i,p_i}(f|_{K_i})$ and its refined value (resulting from the chosen refinement) differ by at least a prescribed tolerance `tol` > 0.

2.3. Smoothness Estimation.

The basic idea presented in the articles [6, 12, 22] is to estimate the regularity of a function to be approximated locally. Then, following along the lines of the hp-approximation approach, if the function is found to be smooth, according to the underlying regularity estimation test, then a p-refinement is performed, otherwise an h-refinement is employed. In [6], the following smoothness indicator, for a (weakly) differentiable function f on an interval K_j, has been introduced (cf. [6, Eq. (3)]):

$$\mathcal{F}_{K_j}[f] := \begin{cases} \dfrac{\|f\|_{L^\infty(K_j)}}{h_j^{-1/2}\,\|f\|_{L^2(K_j)} + \frac{1}{\sqrt{2}} h_j^{1/2}\,\|f'\|_{L^2(K_j)}} & \text{if } f|_{K_j} \not\equiv 0,\\ 1 & \text{if } f|_{K_j} \equiv 0. \end{cases} \tag{F}$$

The motivation behind this definition is the continuous Sobolev embedding $W^{1,2}(K_j) \hookrightarrow L^\infty(K_j)$, which implies that

$$\sup_{v\in H^1(K_j)} \frac{\|v\|_{L^\infty(K_j)}}{h_j^{-1/2}\,\|v\|_{L^2(K_j)} + \frac{1}{\sqrt{2}} h_j^{1/2}\,\|v'\|_{L^2(K_j)}} \le 1;$$

see [6, Proposition 1]. In particular, it follows that $\mathcal{F}_{K_j}[f] \le 1$ in (F); f is classified as being smooth on K_j if $\mathcal{F}_{K_j}[f] \ge \tau$, for a prescribed smoothness testing parameter $0 < \tau < 1$, and nonsmooth otherwise.

To begin, we first consider the special case when f is a polynomial of degree $p_j \ge 1$. Then, the derivative $f^{(p_j-1)}$ of order $p_j - 1$ of f is a linear polynomial, and the evaluation of the smoothness indicator $\mathcal{F}_{K_j}\left[f^{(p_j-1)}\right]$ from (F) is simple to obtain. In fact, let us write $f|_{K_j}$ in terms of a (finite) Legendre series, that is,

$$f|_{K_j}=\sum_{l=0}^{p_j}a_l(\widehat{L}_l\circ\phi_{K_j}^{-1}),\tag{2.8}$$

for coefficients $a_0,\ldots,a_{p_j}\in\mathbb{R}$. Here, $\widehat{L}_l$, $l\geq 0$, are the Legendre polynomials on $[-1,1]$ (scaled such that $\widehat{L}_l(1)=1$ for all $l\geq 0$), and ϕ_{K_j} is the affine scaling of $[-1,1]$ to K_j; cf. (2.3). For f as in (2.8) it can be shown that

$$\mathcal{F}_{K_j}\left[f^{(p_j-1)}\right]=\frac{1+\xi_{p_j}}{\sqrt{1+\frac{1}{3}\xi_{p_j}^2}+\sqrt{2}\xi_{p_j}},\tag{2.9}$$

where $\xi_{p_j}=(2p_j-1)\left|{}^{a_{p_j}}/{}_{a_{p_j-1}}\right|$ (provided that $a_{p_j-1}\neq 0$); see [6, Proposition 3]. In particular, this implies that

$$\frac{1}{2}\approx\frac{\sqrt{3}}{\sqrt{6}+1}\leq\mathcal{F}_{K_j}\left[f^{(p_j-1)}\right]\leq 1;\tag{2.10}$$

cf. [6, §2.2].

In the context of the numerical integration rule (2.2), the above methodology can be adopted as follows: suppose we are given $p_j\geq 2$ quadrature points and weights, $\{\widehat{x}_{p_j,k}\}_{k=1}^{p_j}$ and $\{w_{p_j,k}\}_{k=1}^{p_j}$, respectively. Then,

$$\int_{K_j}f(x)\,\mathsf{d}x\approx Q_{K_j,p_j}(f|_{K_j})=\frac{h_j}{2}\sum_{k=1}^{p_j}w_{p_j,k}(f\circ\phi_{K_j})(\widehat{x}_{p_j,k}).\tag{2.11}$$

We denote the uniquely defined interpolating polynomial of f of degree p_j-1 at the given quadrature points by

$$\Pi_{K_j,p_j-1}f=\sum_{l=0}^{p_j-1}b_l(\widehat{L}_l\circ\phi_{K_j}^{-1}).$$

Due to orthogonality of the Legendre polynomials, we note that

$$b_l=\frac{2l+1}{h_j}\int_{K_j}\Pi_{K_j,p_j-1}f(x)(\widehat{L}_l\circ\phi_{K_j}^{-1})(x)\,\mathsf{d}x,\qquad l=0,\ldots,p_j-1.$$

We further assume that the quadrature rule under consideration is exact for all polynomials of degree up to $2p_j-2$. Thereby,

$$\begin{aligned}b_l&=\frac{2l+1}{2}\sum_{k=1}^{p_j}w_{p_j,k}(\Pi_{K_j,p_j-1}f)\circ\phi_{K_j}(\widehat{x}_{p_j,k})\widehat{L}_l(\widehat{x}_{p_j,k})\\&=\frac{2l+1}{2}\sum_{k=1}^{p_j}w_{p_j,k}(f\circ\phi_{K_j})(\widehat{x}_{p_j,k})\widehat{L}_l(\widehat{x}_{p_j,k}).\end{aligned}$$

Consequently, we infer that

$$\begin{aligned}\xi_{K_j,p_j-1}:&=(2p_j-3)\left|\frac{b_{p_j-1}}{b_{p_j-2}}\right|\\&=(2p_j-1)\frac{\sum_{k=1}^{p_j}w_{p_j,k}(f\circ\phi_{K_j})(\widehat{x}_{p_j,k})\widehat{L}_{p_j-1}(\widehat{x}_{p_j,k})}{\sum_{k=1}^{p_j}w_{p_j,k}(f\circ\phi_{K_j})(\widehat{x}_{p_j,k})\widehat{L}_{p_j-2}(\widehat{x}_{p_j,k})},\end{aligned}\tag{2.12}$$

and thus, in view of (2.9), we use the quantity

$$\mathsf{F}_{K_j,p_j}(f):=\frac{1+\xi_{K_j,p_j-1}}{\sqrt{1+\frac{1}{3}\xi_{K_j,p_j-1}^2}+\sqrt{2}\xi_{K_j,p_j-1}}\in\left(\frac{\sqrt{3}}{\sqrt{6}+1},1\right),\tag{2.13}$$

cf. (2.10), to estimate the smoothness of $f|_{K_j}$. Here, we emphasise that the computation of ξ_{K_j,p_j-1} does not require any additional function evaluations of f since the values $(f\circ\phi_{K_j})(\widehat{x}_{p_j,k})$, $k=1,\ldots,p_j$, have already been determined in the application of the quadrature rule (2.11).

2.4. **Adaptive Variable Order Procedure.** Based on the above derivations, we now propose an *hp*-type adaptive quadrature method. To this end, we start by choosing a tolerance $\texttt{tol} > 0$, a smoothness parameter $\tau \in \left(\sqrt{3}/(\sqrt{6}+1), 1\right)$, and a maximal number $p_{\max} \geq 2$ of possible quadrature points on each subinterval. Furthermore, we define the interval $K_1 = [a, b]$, and a small number p_1, $2 \leq p_1 \leq p_{\max}$, of quadrature points on K_1. Moreover, we initialise the set of subintervals `subs`, the order vector `p` containing the number of quadrature points on each subinterval, and the unknown value `Q` of the integral as follows:

$$\texttt{subs} = \{K_1\}, \qquad \texttt{p} = \{p_1\}, \qquad \texttt{Q} = 0.$$

Then, the basic adaptive procedure is given as follows:

```
1: while subs $\neq \emptyset$ do
2:     [Q1, subs, p] = hprefine($f$, subs, p, $p_{\max}$, $\tau$);
3:     Q = Q + Q1;
4: end while
5: Output Q.
```

Here, `hprefine` is a function, whose purpose is to identify those subintervals in `subs`, which need to be refined further for a sufficiently accurate approximation of the unknown integral. In addition, it outputs a set of subintervals (again denoted by `subs`), as well as an associated order vector (again denoted by `p`) which result from applying the most appropriate refinement, i.e., either *h*- or *p*-refinement as outlined in (i) and (ii) in Section 2.2 above, for each subinterval. Furthermore, `hprefine` returns the sum `Q1` of all quadrature values corresponding to subintervals in the input set `subs` for which no further refinement is deemed necessary. The essential steps are summarised in Algorithm 1.

Algorithm 1 Function [Q, subsnew, pnew] = hprefine(f, subs, p, $p_{\max}$, τ)

```
1: Define subsnew = subs, and pnew = p. Set Q = 0.
2: for each subinterval $K_j \in$ subs do
3:     Evaluate the smoothness indicator $\mathsf{F}_{K_j,p_j}(f)$ from (2.13).
4:     if $\mathsf{F}_{K_j,p_j}(f) < \tau$ then
5:         Apply $h$-refinement to $K_j$, i.e., bisect $K_j$ into two subintervals of equal size and reduce
           the number of quadrature points to $\max(p_j - 1, 1)$ on both of them;
6:         Compute an improved approximation, denoted by $\widetilde{Q}_{K_j}$, of $Q_{K_j,p_j}(f|_{K_j})$ using (2.5)
           on $K_j$.
7:     else if $\mathsf{F}_{K_j,p_j}(f) \geq \tau$ and $p_j + 1 \leq p_{\max}$ then
8:         Apply $p$-refinement to $K_j$, i.e., increase the number of quadrature points to $p_j + 1$
           on $K_j$;
9:         Compute an improved approximation, denoted by $\widetilde{Q}_{K_j}$, of $Q_{K_j,p_j}(f|_{K_j})$ using (2.6)
           on $K_j$.
10:    else if $\mathsf{F}_{K_j,p_j}(f) \geq \tau$ and $p_j + 1 > p_{\max}$ then
11:        Bisect $K_j$ into two subintervals of equal size and retain the number of quadrature
           points $p_j$ on both of them;
12:        Compute an improved approximation, denoted by $\widetilde{Q}_{K_j}$, of $Q_{K_j,p_j}(f|_{K_j})$ using (2.7)
           on $K_j$.
13:    end if
14:    if $|\widetilde{Q}_{K_j} - Q_{K_j,p_j}(f|_{K_j})|$ is sufficiently small then
15:        Update Q = Q + $\widetilde{Q}_{K_j}$;
16:        Eliminate $K_j$ from subsnew and the corresponding entry $p_j$ from pnew.
17:    else
18:        Replace $K_j$ and $p_j$ in subsnew and pnew, respectively, by the corresponding $h$- or
           $p$-refined subintervals as determined above.
19:    end if
20: end for
```

2.5. **Practical Aspects.** In this section we discuss a number of practical issues involved in the implementation of the procedure described in Section 2.4 within a given computing environment.

2.5.1. *Gauss-Quadrature Rules.* In principle, the adaptive procedure presented in Section 2.4 allows for any variable order family of quadrature rules. In our numerical experiments presented in Section 2.6 below, we propose the use of (families of) Gauss-type quadrature schemes. Although they might be criticised for their non-hierarchical structure, in the sense that they require more function evaluations in comparison to more traditional schemes (such as, for example, the adaptive Simpson or fixed-order Gauss-Kronrod rules), our numerical results indicate that their high degree of accuracy may be exploited in a very efficient manner within the *hp*-setting, particularly for smooth functions, with or without locally singular behaviour. Indeed, whilst non-hierarchical lower-order Gauss-type quadrature schemes might not be computationally competitive, it is a well-known feature of *hp*-methods (see, e.g., [18]) that their superiority becomes especially apparent on a variable, higher-order level.

In the current article we employ Gauss-Legendre quadrature points and weights (with at least $p_{\min} = 2$ points and weights); these quantities can be precomputed up to any given order $p_{\max}$ (in practice $p_{\max} = 15$ is usually more than sufficient) or even be generated on the spot in an efficient way (see, e.g., [1,8,21]) if an upper bound $p_{\max}$ cannot be fixed. In addition, we note that the Gauss-Legendre rule based on p points has a degree of exactness of $2p-1$, i.e., the smoothness indicators derived in Section 2.3 can be computed by means of the formula given in (2.12). For a given maximum number $p_{\max}$, we store the points and weights of the Gauss-Legendre rules (on the reference interval $[-1,1]$) with up to $p_{\max}$ points in two $p_{\max} \times (p_{\max} - 1)$-matrices $\boldsymbol{X}$ and $\boldsymbol{W}$, respectively; here, for parameters $p = 2, \ldots, p_{\max}$, the p-th columns of $\boldsymbol{X}$ and $\boldsymbol{W}$ are built from the points and weights of the corresponding p-point Gauss-Legendre quadrature rule, respectively (and complementing the remaining entries in all but the last column by zeros):

$$\boldsymbol{X} = \begin{pmatrix} \widehat{x}_{2,1} & \widehat{x}_{3,1} & \cdots & \widehat{x}_{p_{\max},1} \\ \widehat{x}_{2,2} & \vdots & & \\ & \widehat{x}_{3,3} & & \vdots \\ & \mathbf{0} & \ddots & \\ & & & \widehat{x}_{p_{\max},p_{\max}} \end{pmatrix}, \quad \boldsymbol{W} = \begin{pmatrix} w_{2,1} & w_{3,1} & \cdots & w_{p_{\max},1} \\ w_{2,2} & \vdots & & \\ & w_{3,3} & & \vdots \\ & \mathbf{0} & \ddots & \\ & & & w_{p_{\max},p_{\max}} \end{pmatrix}. \tag{2.14}$$

We note that, for other quadrature rules, the number of rows in the above matrices may be different.

2.5.2. *Vectorised Quadrature.* Following the ideas of [19] we use a vectorised quadrature implementation. This means that, instead of computing the integrals on the subintervals `subs` in Algorithm 1 one at a time, they are all computed at once. This can be accomplished by using fast vector- and matrix-operations, and by carrying out all necessary function evaluations in a single operation by computing the function to be integrated for a vector of input values. Specifically, we write the composite rule

$$I \approx \sum_{K_i \in \texttt{subs}} Q_{K_i,p_i}(f|_{K_i}) = \sum_{K_i \in \texttt{subs}} \frac{h_i}{2} \sum_{k=1}^{p_i} w_{p_i,k} (f \circ \phi_{K_i})(\widehat{x}_{p_i,k})$$

as a dot product of a weight vector $\boldsymbol{w}$ and a function vector $f(\boldsymbol{x})$; here, the former vector contains all (scaled) weights $\{\frac{1}{2} h_i w_{p_i,k}\}_{i,k}$, and the latter vector represents the evaluation of the integrand function f on the vector $\boldsymbol{x}$ of all corresponding quadrature points $\{\phi_{K_i}(\widehat{x}_{p_i,k})\}_{i,k}$ appearing in the sum above. Evidently, these vectors can be built efficiently by extracting (and affinely mapping and scaling) the corresponding rows from the matrices $\boldsymbol{X}$ and $\boldsymbol{W}$ in (2.14). We emphasise that applying vectorised quadrature crucially improves the performance of the overall adaptive procedure (provided that such a technology is available in a given computing environment).

2.5.3. *Smoothness Estimators.* As mentioned before, computing the smoothness indicators from (2.12) does not need any additional function evaluations of the integrand function f; they only require the values of the Legendre polynomials $\widehat{L}_{p-1}$ and $\widehat{L}_{p-2}$ at the points $\{\widehat{x}_{p,k}\}_{k=1}^{p}$, for $p = 2, \ldots, p_{\max}$. These quantities are again precomputable, and can be stored in two matrices

$$\boldsymbol{L}_1 = \begin{pmatrix} L_1(\widehat{x}_{2,1}) & L_2(\widehat{x}_{3,1}) & \cdots & L_{p_{\max}-1}(\widehat{x}_{p_{\max},1}) \\ L_1(\widehat{x}_{2,2}) & \vdots & & \\ & L_2(\widehat{x}_{3,3}) & & \vdots \\ & \mathbf{0} & \ddots & \\ & & & L_{p_{\max}-1}(\widehat{x}_{p_{\max},p_{\max}}) \end{pmatrix}, \tag{2.15}$$

and

$$\boldsymbol{L}_2 = \begin{pmatrix} L_0(\widehat{x}_{2,1}) & L_1(\widehat{x}_{3,1}) & \cdots & L_{p_{\max}-2}(\widehat{x}_{p_{\max},1}) \\ L_0(\widehat{x}_{2,2}) & \vdots & & \\ & L_1(\widehat{x}_{3,3}) & & \vdots \\ & \mathbf{0} & \ddots & \\ & & & L_{p_{\max}-2}(\widehat{x}_{p_{\max},p_{\max}}) \end{pmatrix}. \tag{2.16}$$

Then, the sums in (2.12) are vectorised similarly as described above. In particular, the computation of the smoothness estimators can be undertaken with an almost negligible computational cost.

2.5.4. *Stopping Criterion.* In order to implement the stopping-type criterion in line 14 of Algorithm 1, we exploit an idea that was proposed in the context of adaptive Simpson quadrature in [7]. More precisely, given a possibly rough approximation $\texttt{iguess} \approx \int_a^b f(x)\,\mathsf{d}x$ of the unknown integral I from (2.1) (e.g., obtained from a Monte-Carlo calculation such that both the approximation and the exact value are of the same magnitude; cf. [7]), and a tolerance $\texttt{tol} > 0$, we redefine

$$\texttt{iguess} = \texttt{iguess} * \texttt{tol}/\texttt{eps};$$

here, `eps` represents the smallest (positive) machine number in a given computing environment. Then, using the comparison operator ==, we accept the difference $|\widetilde{Q}_{K_j} - Q_{K_j,p_j}(f|_{K_j})|$ to be sufficiently small with respect to the given tolerance `tol` if the logical call

$$\texttt{iguess} + |\widetilde{Q}_{K_j} - Q_{K_j,p_j}(f|_{K_j})| == \texttt{iguess};$$

yields a `true` value.

2.6. **Numerical Examples.** In order to test our approach, we consider a number of benchmark problems on the interval $[0, 1]$. Specifically, the following functions will be studied:

$$\begin{aligned} f_1(x) &= \exp(x), \\ f_2(x) &= \sqrt{|x - 1/3|}, \\ f_3(x) &= \operatorname{sech}(10(x - 1/5))^2 + \operatorname{sech}(100(x - 2/5))^4 \\ &\quad + \operatorname{sech}(1000(x - 3/5))^6 + \operatorname{sech}(1000(x - 4/5))^8, \\ f_4(x) &= \cos(1000x), \\ f_5(x) &= \begin{cases} 0 & \text{if } x \leq 1/3, \\ 1 & \text{if } x > 1/3. \end{cases} \end{aligned}$$

Whilst the first function, f_1, is analytic, the second function, f_2, is smooth except at $1/3$ (see Figure 1 (top)). Furthermore, f_3 was proposed in [10] in the context of the chebfun package [11]; this is a smooth function that exhibits several very thin spikes (see Figure 2 (top)). Moreover, f_4 is highly oscillating, and f_5 is an example of a discontinuous function.

	hp-adapt. quad.			*adapt. Simpson quad.*
	# fct. calls	# sing. fct. ev.	cpu [sec]	# sing. fct. ev.
f_1	52	9	0.0031	4,096
f_2	1,718	65	0.0224	25,488
f_3	2,427	33	0.0144	72,528
f_4	50,534	35	0.0180	1,965,376
f_5	1,273	106	0.0342	784

Table 1. Performance data for *hp*-type adaptive quadrature.

We perform our computations in Matlab[1] on a single 2.6GHz processor. The tolerance is set to $\texttt{tol} = 0.3 \times 10^{-15}$ (which is close to machine precision in Matlab), the smoothness estimation parameter is prescribed as $\tau = 0.6$, and $p_{\max} = 15$. Within this setting, the adaptive procedure generates results that are accurate to machine precision, for all of the considered examples. In Table 1, for each of the functions $f_1, \ldots, f_5$ above, we present the number of function calls (# fct. calls) in the vectorised quadrature implementation (counting a single application of the integrand function to a vector input as 1; cf. Section 2.5.2), as well as the number of single function evaluations (# sing. fct. ev.) taking into account the number of scalar entries of a vector input in each function call. The latter number is compared with the number of scalar function evaluations performed in a classical adaptive Simpson procedure as proposed in [7] (which is based on employing the two end points as well as the midpoint on each subinterval, and reuses the former two points without recomputing). Except for the last function, f_5, where a low-order quadrature rule is more effective, the remarkable efficiency of the proposed *hp*-type quadrature becomes clearly visible. This is confirmed with the expeditious cpu times (which do not include the computation of the precomputable matrices $\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{L}_1, \boldsymbol{L}_2$ from (2.14), (2.15), and (2.16)) for each of the examples.

In order to illustrate how the *hp*-adaptive procedure performs, we depict the final *hp*-mesh for f_2 and f_3 in Figure 1 (bottom) and Figure 2 (bottom), respectively. Here, along the horizontal axis we present the subintervals obtained as a result of the adaptive process, and on the vertical axis the number of quadrature points introduced on each subinterval is displayed. In both examples, we see that smooth regions in the underlying integrand are resolved by employing larger subintervals featuring a higher number of quadrature points, whereas close to singularities, the number of quadrature points is kept low on very small integration subdomains. It is noteworthy that this behaviour is well-known from *hp*-finite element methods for differential equations, where high-order algebraic or even exponential convergence rates can be obtained by applying this type of *hp*-refinement procedure; see [18] for details.

3. Conclusions

In this article we proposed a new adaptive quadrature strategy, which features both local subdivision of the integration domain, as well as local variation of the number of quadrature points employed on each subinterval. Our approach is inspired by the *hp*-adaptive finite element methodology based on *hp*-adaptive smoothness testing. In combination with a vectorised quadrature implementation, the proposed adaptive quadrature algorithm is able to deliver highly accurate results in a very efficient manner. Since our approach is closely related to the *hp*-finite element technique, it can be extended to multiple dimensions, including, in particular, the application of anisotropic refinements of the underlying domain of integration, together with the exploitation of different numbers of quadrature points in each coordinate direction on each subinterval (based, for example, on anisotropic Sobolev embeddings as outlined in [6, §3.1]).

[1]The MathWorks, Inc.

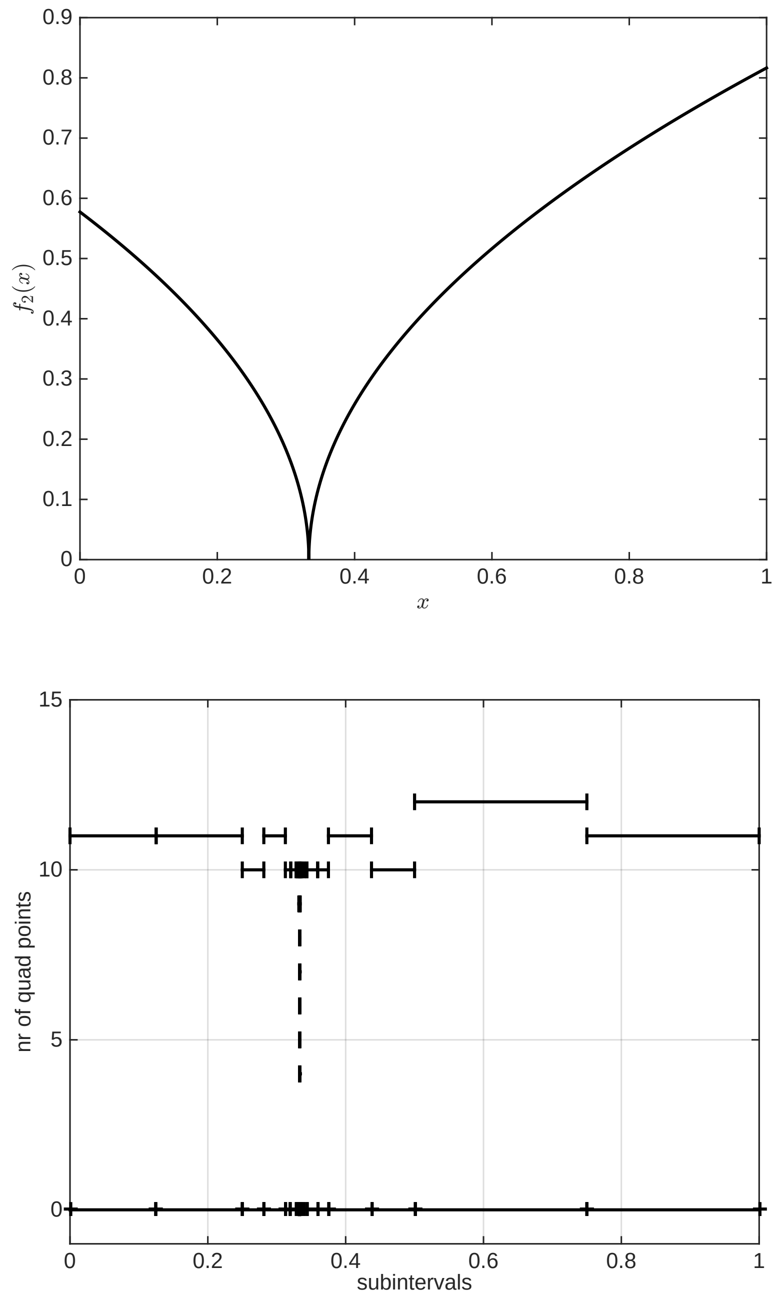

FIGURE 1. Function f_2: Graph (top) and *hp*-mesh (bottom).

REFERENCES

1. C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral methods in fluid dynamics*, Springer Series in Computational Physics, Springer-Verlag, New York, 1988.
2. G. Dahlquist and Å. Björck, *Numerical methods in scientific computing. Vol. I*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.
3. P. J. Davis and P. Rabinowitz, *Methods of numerical integration*, Dover Publications, Inc., Mineola, NY, 2007, Corrected reprint of the second (1984) edition.
4. L. Demkowicz, *Computing with hp-adaptive finite elements. Vol. 1*, Chapman & Hall/CRC Applied Mathematics and Nonlinear Science Series, Chapman & Hall/CRC, Boca Raton, FL, 2007, One and two dimensional elliptic and Maxwell problems.

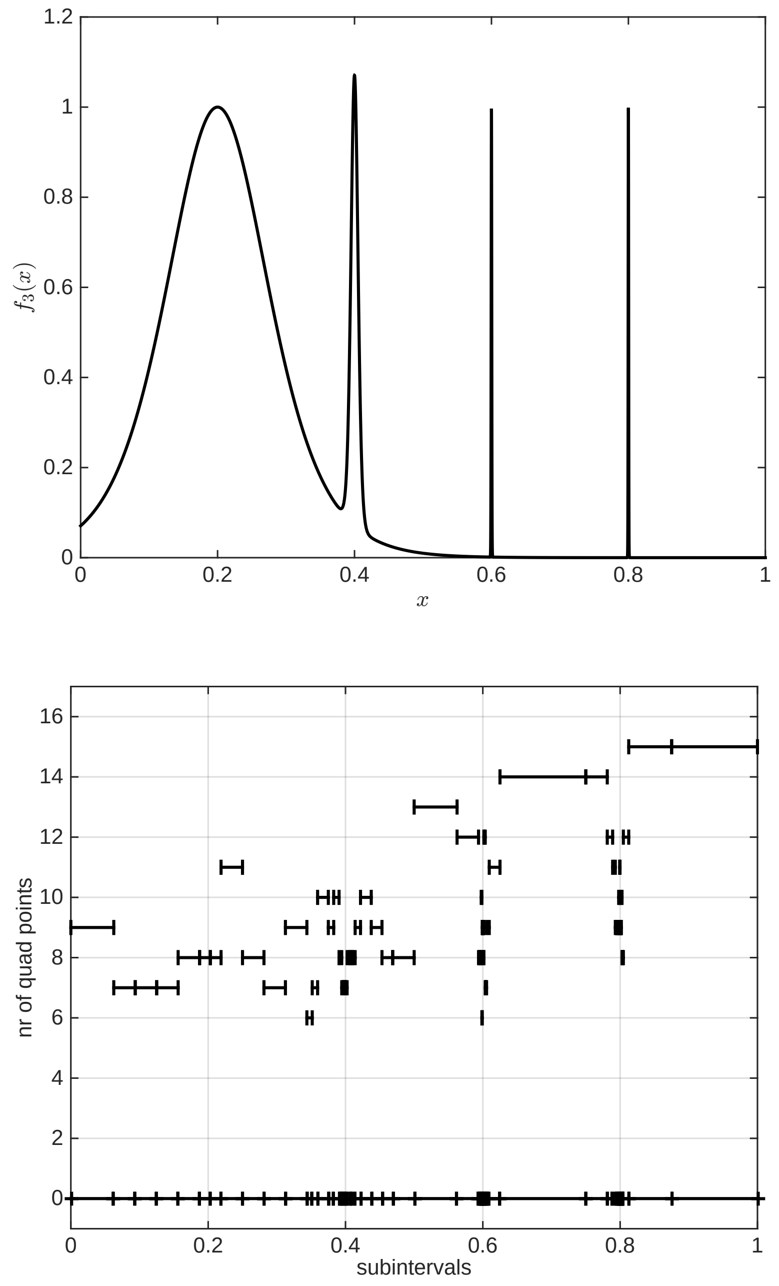

FIGURE 2. Function f_3: Graph (top) and hp-mesh (bottom).

5. R. DeVore and K. Scherer, *Variable knot, variable degree spline approximation to x^β*, Quantitative approximation (Proc. Internat. Sympos., Bonn, 1979), Academic Press, New York-London, 1980, pp. 121–131.
6. T. Fankhauser, T. P. Wihler, and M. Wirz, *The hp-adaptive FEM based on continuous Sobolev embeddings: isotropic refinements*, Computers & Mathematics with Applications. An International Journal **67** (2014), no. 4, 854–868.
7. W. Gander and W. Gautschi, *Adaptive quadrature—revisited*, BIT **40** (2000), no. 1, 84–101.
8. A. Glaser, X. Liu, and V. Rokhlin, *A fast algorithm for the calculation of the roots of special functions*, SIAM Journal on Scientific Computing **29** (2007), no. 4, 1420–1438.
9. W. Gui and I. Babuška, *The h, p and $h-p$ versions of the finite element method in one-dimension, parts I–III*, Numer. Math. **49** (1986), no. 6, 577–683.
10. N. Hale, *Spike integral*, 2010, http://www.chebfun.org/examples/quad/SpikeIntegral.html.

11. N. Hale and L. N. Trefethen, *Chebfun and numerical quadrature*, Science China Mathematics **55** (2012), no. 9, 1749–1760.
12. P. Houston and E. Süli, *A note on the design of hp–adaptive finite element methods for elliptic partial differential equations*, Comput. Methods Appl. Mech. Engrg. **194(2-5)** (2005), 229–243.
13. J. M. Melenk and B. I. Wohlmuth, *On residual-based a posteriori error estimation in hp-FEM*, Adv. Comp. Math. **15** (2001), 311–331.
14. W. F. Mitchell and M. A. McClain, *A comparison of hp-adaptive strategies for elliptic partial differential equations*, ACM Transactions on Mathematical Software (TOMS) **41** (2014), 2:1–2:39.
15. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes*, third ed., Cambridge University Press, Cambridge, 2007, The art of scientific computing.
16. K. Scherer, *On optimal global error bounds obtained by scaled local error estimates*, Numer. Math. **36** (1980/81), no. 2, 151–176.
17. D. Schötzau, C. Schwab, and T. P. Wihler, *hp-DGFEM for second-order mixed elliptic problems in polyhedra*, Math. Comp. (in press).
18. C. Schwab, *p- and hp-FEM – Theory and application to solid and fluid mechanics*, Oxford University Press, Oxford, 1998.
19. L. F. Shampine, *Vectorized adaptive quadrature in Matlab*, J. Comput. Appl. Math. **211** (2008), no. 2, 131–140.
20. P. Solin, K. Segeth, and I. Dolezel, *Higher-order finite element methods*, Studies in advanced mathematics, Chapman & Hall/CRC, Boca Raton, London, 2004.
21. J. Waldvogel, *Fast construction of the Fejér and Clenshaw-Curtis quadrature rules*, BIT. Numerical Mathematics **46** (2006), no. 1, 195–202.
22. T. P. Wihler, *An hp-adaptive strategy based on continuous Sobolev embeddings*, J. Comput. Appl. Math. **235** (2011), 2731–2739.

School of Mathematical Sciences, University of Nottingham, University Park, Nottingham, NG7 2RD, UK
E-mail address: `Paul.Houston@nottingham.ac.uk`

Mathematisches Institut, Universität Bern, Sidlerstrasse 5, CH-3012 Bern, Switzerland
E-mail address: `wihler@math.unibe.ch`